A Cascaded Unsupervised Model for PoS Tagging

NECVA BÖLÜCÜ, Hacettepe University, Turkey
BURCU CAN, Research Institute in Information and Language Processing, University of Wolverhampton, UK

Part of speech (PoS) tagging is one of the fundamental syntactic tasks in Natural Language Processing (NLP), that assigns a syntactic category to each word within a given sentence or context (such as noun, verb, adjective etc). Those syntactic categories could be used to further analyze the sentence-level syntax (e.g. dependency parsing) and thereby extract the meaning of the sentence (e.g. semantic parsing). Various methods have been proposed for learning PoS tags in an unsupervised setting without using any annotated corpora. One of the widely used methods for the tagging problem is log-linear models. Initialization of the parameters in a log-linear model is very crucial for the inference. Different initialization techniques have been used so far. In this work, we present a log-linear model for PoS tagging that uses another fully unsupervised Bayesian model to initialize the parameters of the model in a cascaded framework. Therefore, we transfer some knowledge between two different unsupervised models to leverage the PoS tagging results, where a log-linear model benefits from a Bayesian model's expertise. We present results for Turkish as a morphologically rich language and for English as a comparably morphologically poor language in a fully unsupervised framework. The results show that our framework outperforms other unsupervised models proposed for PoS tagging.

CCS Concepts: • **Mathematics of computing → Probability and statistics**; **Probabilistic inference problems**; **Probabilistic reasoning algorithms**; • **Computing methodologies → Natural language processing**; **Phonology / morphology**; **Unsupervised learning**;

Additional Key Words and Phrases: Unsupervised learning; Part-of-Speech tagging (PoS tagging); Log-linear model; Bayesian learning

ACM Reference format:
Necva Bölücü and Burcu Can. 2020. A Cascaded Unsupervised Model for PoS Tagging. 1, 1, Article 1 (October 2020), 23 pages.
https://doi.org/10.1145/nnnnnnn.nnnnnnn

1 INTRODUCTION

Part-of-speech (PoS) tagging also called grammatical tagging is the task of assigning a syntactic category to each word in a given context (i.e. a sentence or a phrase). This task is one of the long-standing problems in computational linguistics and one of the preliminary tasks in Natural Language Processing (NLP) since many other applications require syntactic categories of words to further analyze the meaning of a sentence. For example, it would be easier to extract the meaning of the word *saw* if its syntactic category is known a priori. So any improvement on the accuracy of PoS tagging leads to a significant impact on other NLP applications such as machine translation [53] and dependency parsing [15]. Various machine learning methods have already been applied to PoS tagging, either supervised or unsupervised. However, available annotated datasets for the NLP tasks generally do not contain PoS tag information, especially in resource

scarce languages. At this point, unsupervised learning has taken to the stage in NLP since it reduces the annotation cost without requiring annotated corpora.

Unsupervised learning has been applied in almost any field of computational linguistics (e.g. morphology [38], dependency parsing [29]). Most of the unsupervised models are statistical (Bayesian [24], log-linear [48]). Recently, also deep learning models have been proposed in such unsupervised frameworks [40]. Regardless of the type of the model, the main goal is to learn a set of model parameters that maximizes a predefined objective function that usually defines the likelihood of data under a hidden structure, i.e. PoS tagging. However, dealing with sentences and learning the syntactic structure of sentences have been a challenging problem due to the large space of hidden parameters within the syntactic structure of a language. The space is even larger in free-order languages such as Turkish.

Log-linear models have been widely used in various NLP tasks including PoS tagging. However parameter estimation problem exists for PoS tagging alike other tasks due to a large search space especially when there is no prior information such as the distribution of words across different syntactic categories. To overcome this problem, many parameter estimation techniques have been used such as expectation-estimation (EM) [30], contrastive estimation [48, 49]. These techniques are all *local search* techniques that improve the parameters through hill-climbing [26]. Another common way to overcome the sparsity problem in tagging is to reduce the search space by using a tagging dictionary that consists of words and their possible PoS tags that are obtained from a labeled dataset. However, if there is not any annotated data, then different methods are required to tackle the sparsity problem.

The popular method used in log-linear models for PoS tagging in an unsupervised setting is contrastive estimation. In contrastive estimation, if there is not enough labeled data, it is enlarged by generating negative samples (that are not seen in the dataset and also very unlikely to be seen in the language) using the positive samples (that are already in the dataset). Therefore, the probability of the correct ones are increased while decrementing the probability of the incorrect ones, which is proposed by Smith and Eisner [48]. For the PoS tagging task, the correct samples are the valid sentences that are already given in the labeled dataset and the negative samples are the incorrect/invalid sentences that are derived by using the sentences in the dataset by applying some perturbations. For example, using the positive sentence *I am reading a book*, the negative sentences such as *I am reading book*, *I reading am book*, *I am book reading* are generated.

The main problem of estimation methods used in log linear models is that such a log linear model with hidden variables is not globally concave and eventually it gets stuck in local optima. As Smith and Eisner [48] describe, the bias in the initialization of the parameters will affect the quality of the estimate and the performance of the method. An algorithm such as EM will find numerous solutions in case of random initialization of the parameters [14, 49]. Various solutions are applied to find the global optimum solution, such as heuristic initialization of the model parameters [50], random restarts [47], or annealing [47, 49]. Here we introduce the idea of initializing the parameters in a log-linear model in a cascaded framework by using another unsupervised model to obtain the initial parameters. This will not guarantee a global solution, but will give a better convergence of the parameters compared to a random initialization.

In this article, we focus on a fully unsupervised log-linear model that does not make use of an external tagging dictionary and therefore an annotated dataset. We focus on the initialization of the parameters without using any prior supervised knowledge in contrast to other PoS tagging models [24, 43, 48] that use a dictionary and therefore an available annotated dataset in a supervised setting.

We propose a cascaded model that contains two different unsupervised models to enhance the quality of the estimate without using any external knowledge in a fully unsupervised framework. The idea behind combining different models in a cascaded learning framework is to compensate the weakness of a mathematical model with another mathematical

model, thereby each model benefits from other model's expertise because each mathematical model can define different sets of features and different mathematical functions over those features. Cascade learning was first introduced in classification related problems such as visual object detection [8, 12, 44]. How this cascaded framework helps in improving each model's performance is explained in different studies in the literature. For example, Heitz et al. [28] state that information is shared between different components and therefore state-of-the-art methods benefit from each other's expertise and in their work a cascaded classification model is introduced. Similarly, they also train independent classifiers on the same training data. Brubaker et al. [12] also apply cascade learning for boosted ensembles and each stage is trained to reject the false positives of previous stages, therefore the overall detection capability improves in each stage of the cascade. Recently, also deep cascaded models have been introduced with a similar idea, where each high-level neural network component receives the result from a low-level neural network architecture [25, 55].

Our algorithm first uses the fully unsupervised Bayesian model proposed by Goldwater and Griffiths [24] to tag an unlabeled dataset, that initiates the first step of the algorithm. However, in contrast to the original model we do not make use of any tagging dictionary. The parameters of a log-linear model based on contrastive estimation proposed by Smith and Eisner [48] are initialized using the tags learned in the first step of the algorithm. Therefore, we transfer some knowledge between two unsupervised models to leverage the performance of both. Unlike the second model, we again do not make use of any tagging dictionary, therefore the model is trained without using any annotated data. Therefore, the log-linear model benefits from the Bayesian model's expertise and eventually improves the tagging accuracy. However, differently from a typical cascaded framework where each stage is applied subsequently for the final prediction of the model, here we only use the final stage for producing the testing result. In other words, each cascade component is affected from each other only during training.

We show that the log-linear model's accuracy is improved significantly by using another Bayesian model for the initialization of its parameters. This indicates that the initialization of log-linear parameters is crucial in learning.

Therefore, we have two contributions to the field with this study. One of them is to show that using two different unsupervised models by transferring some knowledge between the two models makes a significant improvement on the quality of the estimate. The second contribution is related to the quality of the estimate in log-linear models by reforming the initialization of parameters. We show that using another unsupervised model reduces the search space and therefore it improves the quality of the estimate and thereby the performance. We apply these two approaches for the PoS tagging task and we report competitive results compared to other baseline models such as the original log-linear model [48] and the fully Bayesian model [24], or even one of the best performing PoS tagging algorithms, Brown Clustering [11].

The paper is structured as follows: Section 2 addresses the related work on PoS tagging task, section 3 describes the proposed cascaded model with the two components, log-linear model and fully unsupervised Bayesian model, section 4 presents the experimental results obtained from the proposed model, and finally section 5 concludes the paper with the future goals.

2 RELATED WORK

Various machine learning methods have been applied to PoS tagging, those include supervised, unsupervised, or semi-supervised methods. Some of those methods have seen the problem as a classification/clustering task [11, 18], and some of them have tackled the tagging as a sequence labeling problem [4, 24, 30].

Unsupervised PoS tagging does not require any annotated corpus, but instead computational methods are introduced to induce the syntactic labels [11, 30] in such learning schemes. On the other hand, supervised PoS tagging generally

relies on annotated corpus to obtain some prior knowledge for learning. For example some models make use of labelled corpus to obtain a tagging dictionary [37, 42], some of them exploit the word/tag frequencies [14], and some others extract a rule set from the corpus that defines the tagging rules [10] to be applied during learning.

Since our main focus is unsupervised learning, we will address only the unsupervised studies on PoS tagging here.

One of the oldest unsupervised algorithms proposed for PoS tagging is Brown clustering [11]. In Brown clustering, a class-based n-gram model is used to cluster words using their distributional features in various contexts. Sparsity due to using actual word tokens is reduced by using the classes of the contextual words rather than using the word tokens. Once word clusters are learned by maximum likelihood estimation using a language model, clusters are merged further based on the average mutual information between clusters to build a hierarchical clustering tree. The results show that clusters correspond to both syntactic and semantic classes since semantically related words also occur in similar contexts likewise syntactically similar words. Similarly, Schütze [45] uses two previous and two following words as contextual features to cluster words into syntactic categories. Singular Value Decomposition (SVD) [20] and Buckshot clustering [19] algorithms are applied consecutively to learn the categories by reducing the dimensions.

Biemann [6] propose a novel clustering algorithm called Chinese Whispers that is a graph-based clustering algorithm. The algorithm uses a fixed context window and the most frequent words in the context windows as features, where each word corresponds to a node in the graph and edges correspond to their neighbour relations with other words.

Johnson [30] adopts a Bayesian framework within a Hidden Markov Model (HMM) and compares various estimators for the inference and concludes that using Expectation-Maximization (EM) in HMMs produces poor results because the estimator assigns equal number of words to each state (i.e. tag), where the actual distribution is highly skewed. In other words, some syntactic categories have more words compared to other syntactic categories (i.e. open and close classes). Johnson [30] uses a Bayesian perspective instead by assigning priors to the tag transitions and emissions in an HMM, which improves the results of the tagger. Goldwater and Griffiths [24] propose another Bayesian approach that is built upon a second order HMM with symmetric Dirichlet priors over transition and emission distributions and use Gibbs sampling to estimate the parameters. We utilize the Bayesian model of Goldwater and Griffiths [24] in our cascaded framework to initialize the parameters of another unsupervised model.

Smith and Eisner [48] propose a log-linear model with contrastive estimation that augments the dataset with negative examples. To this end, the authors apply some perturbation methods such as deletion of a word from a sentence or swapping two adjacent words in a sentence. Therefore, the likelihood of data is maximized by shifting some of the probability mass from the negative samples (that are unlikely to be observed in the language) to the positive examples. We are motivated by this model in our work and we will describe it in detail in Section 3.

A more recent work is by Berg-Kirkpatrick et al. [5] who also use a log-linear model. The authors use morphological features in their sequence model to induce the words that share the same morphological features that will be assigned to similar syntactic categories. Clark [16] proposes another model that also combines the distributional and morphological information to learn the syntactic categories.

Stratos et al. [51] introduce a new model called Anchor HMMs, where each HMM tag is matched to at least one word that can have no other tag in an HMM. For instance, word *the* can appear only as a determiner tag and cannot be associated with another tag. The authors use negative matrix factorization (NMF) [3] algorithm, which was first introduced for topic modeling to learn the parameters from unlabeled data. Here, they propose to use NMF for sequence labelling task.

Recently, a variety of neural network models have been proposed for various NLP tasks. PoS tagging is also one of these problems that has been exposed to deep neural networks. Collobert et al. [17] propose a multi-layer neural network

architecture for different NLP tasks such as PoS tagging, named-entity recognition (NER), semantic role labelling. The goal of this study and of many others is to avoid task-specific features, which are extracted by the neural network automatically. Wang et al. [54] train a bidirectional Long Short Term Memory Network (biLSTM) for PoS tagging as a sequence labeling problem without using any morphological features to make the model applicable on languages that lack of morphological knowledge. Labeau et al. [31] propose a model that learns word representations through a convolutional neural network (CNN) using only the character stream of words. Once the word representations are learned, the PoS tags are predicted by Viterbi algorithm. Plank et al. [41] present another neural network architecture using biLSTMs that combines PoS tagging loss function with an auxiliary loss function for rare words. The authors evaluate the model using word, character and byte embeddings for PoS tagging on 22 languages. The model obtains the state-of-art performance especially across morphologically rich languages. Andor et al. [1] introduce a globally normalized transition-based neural network that achieves comparable results on three tasks: PoS tagging, dependency parsing and sentence compression. However, it is worth mentioning that all of the current neural network based models propose a supervised learning framework.

Most of the previous statistical methods proposed for PoS tagging [4, 21, 24, 30] use dictionary constraints to lead the algorithms, that hinders the models from being completely unsupervised. In our work, we combine two different unsupervised models to leverage the performance of PoS tagging in a fully unsupervised setting. To this end, we transfer some knowledge between two unsupervised models by applying cascaded learning in a fully unsupervised setting which makes our model different than others.

3 CASCADED POS TAGGING FRAMEWORK

Our unsupervised cascaded PoS tagging model involves two different unsupervised tagging components that are unified in a single framework with some shared knowledge. The first model is a Bayesian model that is motivated by the model proposed by Goldwater and Griffiths [24] and the second model is based on the log linear model proposed by Smith and Eisner [48].

Since both models use a tagging dictionary, here we aim to eliminate the usage of a tagging dictionary and therefore learn PoS tags in a fully unsupervised setting. Here, we train the Bayesian model first and use the predicted PoS tags to initialize the parameters of the log linear model. Both models are described thoroughly below, which is followed by the description of the overall algorithm.

3.1 1st Step: Bayesian Model for PoS tagging

We adopt the Bayesian HMM model proposed by Goldwater and Griffiths [24] for the first component of our cascaded model. However, we modify the model in a fully unsupervised setting without using any labeled data.

In the original version of the Bayesian model, the authors use a tagging dictionary where each word and its possible tags are available. Therefore, the model is only trained for learning to choose the correct tag from the tagging dictionary which allows only certain tags for each word. Although, we adopt their mathematical model, training is performed differently to build a fully unsupervised setting.

3.1.1 Mathematical Model Definition. The Bayesian HMM model is an extension of a standard HMM model from a Bayesian perspective. A standard HMM model normally seeks for the latent variables that maximize the underlying probability distributions (i.e. transition and emission). Either maximum likelihood (ML) values with the corpus statistics are used for the estimation of the parameters, or maximum a posteriori (MAP) solutions that incorporate prior

information on the parameters during estimation are used in such a standard non-Bayesian model. However, in a Bayesian framework, we aim to learn the distribution over the latent variables rather than learning the parameters that maximize the given probability distribution. Therefore, those parameters are normally integrated out to have a summary over the distribution which has all possible values of the parameters rather than a single point estimate of each parameter.

The Bayesian mathematical model used in this work is defined as follows [24]:

$$t_i|t_{i-1}, t_{i-2} = t^{'}, \tau^{(t,t^{'})} \propto Mult(\tau^{(t,t^{'})}) \quad (1)$$

$$w_i|t_i = t, \omega^{(t)} \propto Mult(\omega^{(t)}) \quad (2)$$

$$\tau^{(t,t^{'})}|\alpha \propto Dirichlet(\alpha) \quad (3)$$

$$\omega^{(t)}|\beta \propto Dirichlet(\beta) \quad (4)$$

where w_i denotes the ith word and t_i is its PoS tag. $Mult(\omega^t)$ is the emission distribution in the form of a Multinomial distribution with parameters $\omega^{(t)}$. The Multinomial parameters are generated by $Dirichlet(\beta)$ with hyperparameters β. Analogously, $Mult(\tau^{(t,t^{'})})$ is the tag transition distribution with parameters $\tau^{(t,t^{'})}$ that are also generated by $Dirichlet(\alpha)$ with hyperparameters α. The plate diagram of the model is given in Figure 1.

Based on the mathematical model, the conditional probabilities used for the inference of the model are defined as follows [24]:

$$P(t_i|\mathbf{t_{-i}}, \alpha) = \frac{n_{(t_{i-2}, t_{i-1}, t_i)} + \alpha}{n_{(t_{i-2}, t_{i-1})} + T\alpha} \quad (5)$$

$$P(w_i|\mathbf{t_{-i}}, \mathbf{w_{-i}}, \beta) = \frac{n_{(t_i, w_i)} + \beta}{n_{(t_i)} + W_{t_i}\beta} \quad (6)$$

Here $\mathbf{t_{-i}}$ denotes the current values of all tags except t_i, $\mathbf{w_{-i}}$ denotes the complete word list excluding w_i, W_{t_i} is the number of word types in the corpus, T is the size of the tag set, n_{t_i} is the number of words tagged with t_i, $n_{(t_i, w_i)}$ is the number of tag-word pairs (t_i, w_i), $n_{(t_{i-2}, t_{i-1})}$ is the frequency of the tag bigram $< t_{i-2}, t_{i-1} >$, and finally $n_{(t_{i-2}, t_{i-1}, t_i)}$ is the frequency of the tag trigram $< t_{i-2}, t_{i-1}, t_i >$. As noted before, the parameters τ and ω do not exist in Equations 5 and 6 since they are integrated out. Therefore, the distributions $P(t_i|\mathbf{t_{-i}}, \alpha)$ and $P(w_i|\mathbf{t_{-i}}, \mathbf{w_{-i}}, \beta)$ contain a distribution over their parameters τ and ω, which are not assigned a single value with a point estimate method (such as ML or MAP).

The inference involves estimating the following posterior distribution, which is the conditional probability of tags given all the words in the dataset:

$$P(\mathbf{t}|\mathbf{w}, \alpha, \beta) \propto P(\mathbf{w}|\mathbf{t}, \beta)P(\mathbf{t}|\alpha) \quad (7)$$

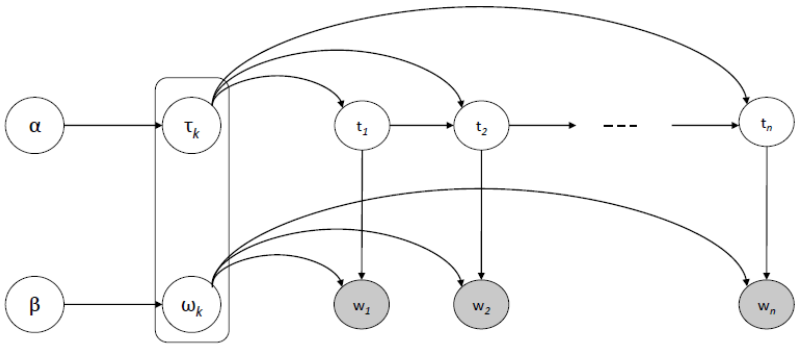


Fig. 1. The plate diagram of the Bayesian HMM.

We use Gibbs sampling for the inference. All tags are randomly initialized and a tag is sampled for each word from the following sampling distribution in each step [24]:

$$
\begin{aligned}
P(t_i|\mathbf{t_{-i}}, \mathbf{w_{-i}}, \alpha, \beta) \quad = \quad & \frac{n_{(t_i,w_i)}+\beta}{n_{t_i}+W_{t_i}\beta} \cdot \frac{n_{(t_{i-2},t_{i-1},t_i)}+\alpha}{n_{(t_{i-2},t_{i-1})}+T\alpha} \\
\cdot \quad & \frac{n_{(t_{i-1},t_i,t_{i+1})}+I(t_{i-2}=t_{i-1}=t_i=t_{i+1})+\alpha}{n_{(t_{i-1},t_i)}+I(t_{i-2}=t_{i-1}=t_i)+T\alpha} \\
\cdot \quad & \frac{n_{(t_i,t_{i+1},t_{i+2})}+I(t_{i-2}=t_i=t_{i+2},t_{i-1}=t_{i+1})+I(t_{i-1}=t_i=t_{i+1}=t_{i+2})+\alpha}{n_{(t_i,t_{i+1})}+I(t_{i-2}=t_i,t_{i-1}=t_{i+1})+I(t_{i-1}=t_i=t_{i+1})+T\alpha}
\end{aligned} \tag{8}
$$

where $I(.)$ is an identity function that gives 1 if its argument is true, and otherwise 0. Sampling a tag affects three trigrams. Therefore, those changes are taken into account with the identity function.

3.2 2nd Step: Log-Linear Model for PoS Tagging

3.2.1 Mathematical Model Definition. Let s_i be a sentence where $D = \{s_1, s_2, \cdots, s_m\}$ is the dataset that involves m sentences and y be any hidden PoS tag sequence. Each sentence is defined as $s_1 = \{w_1, w_2, \cdots, w_m\}$ where w_j is the jth word in the sentence. The log likelihood of the dataset under the parameters θ is given as follows [48]:

$$L_N(\theta) = \log \prod_{s_i \in D} p(s_i) \tag{9}$$

$$= \log \prod_{s_i \in D} \sum_{y \in Y} p(s_i, y) \tag{10}$$

$$= \log \prod_{s_i \in D} \frac{\sum_{y \in Y} u(s_i, y|\vec{\theta})}{\sum_{(s^*,y^*) \in N(s_i) \times Y} u(s^*, y^*|\vec{\theta})} \tag{11}$$

where Y denotes the set of unobserved hidden structures, that are the PoS tag sequences y^* assigned for the negative sentences s^* generated from the original sentence s_i. Here, $u(s_i, y|\vec{\theta})$ is the unnormalized probability of $s(i)$ under the

tag sequence y. The unnormalized probability of s_i is defined as a log-linear model as follows:

$$u(s_i|\vec{\theta}) = \sum_{y \in Y} u(s_i, y|\vec{\theta}) \tag{12}$$

$$= \sum_{y \in Y} exp(\vec{\theta} \cdot \vec{f}(s_i, y)) \tag{13}$$

where $\vec{f}(s_i, y)$ denotes the feature vector of s_i tagged with y, and θ is the corresponding weight vector that consists of the weights assigned for each feature. Here we use the basic features as suggested by [5]. These features are the transition (tag bigrams) and emission features. An example to transition and emission features is given below:

$$f_1(h, y) = \begin{cases} 1 & if\ y_j = t_1\ and\ y_j + 1 = t_2\ for\ some\ j \in 1\ to\ n \\ 0 & otherwise \end{cases} \tag{14}$$

$$f_2(h, y) = \begin{cases} 1 & if\ y_j = t_1\ and\ w_j = \text{“}model\text{”}\ for\ some\ j \in 1\ to\ n \\ 0 & otherwise \end{cases}$$

Here, h is a history where it could be either a tag or a preceding word. For example, f_1 is a tag transition feature where the history refers to t_1 and f_2 is a word emission feature and the history is the word *model*. The value of each observed feature is 1. Here, n denotes the time sequence within the data, i.e. the order of the current word in the data.

We apply contrastive estimation to normalize the probability of each sentence, where the normalization is performed through the negative samples of the sentence, $N(s_i)$ that also includes the sentence itself. DEL1WORD, TRANS1 and DEL1SUBSEQ are used for perturbation. DEL1WORD deletes a word in a sentence, TRANS1 swaps two words in a sentence, and DEL1SUBSEQ deletes a word sequence within a sentence. Lattice structures are used for the perturbation operations. One of the lattices is built for the original sentence and the others correspond to the negative perturbations of the original sentence. An example lattice representation of the sentence *“All came from Cray Research"* is given in Figure 2.

Since we do not know the actual tags of the words, we can only estimate the probabilities of the original sentence and negative sentences from the expected counts that can be computed by dynamic programming. We use inside-outside algorithm to estimate the probabilities. The probability of each sentence is estimated using inside-outside algorithm as follows:

$$u(s_i|\vec{\theta}) = \sum_{y \in Y} u(s_i, y|\vec{\theta}) \tag{15}$$

$$= \sum_{j=1}^{N} \alpha_T(j) \tag{16}$$

where $\alpha_T(j)$ denotes the forward probability ending in state j ($PoS\ tag\ j$) at time T ($the\ Tth\ word$). The probability of each possible tag sequence for the given sentence is summed over by estimating the probability of each state ending in any N possible PoS tags in the model. The probability of being in state j at time t after observing the words $w_1, w_2, \cdots, w_t$ is:

$$\alpha_t(j) = p(w_1, w_2, \cdots, w_t, y_t = j|\vec{\theta}) \tag{17}$$

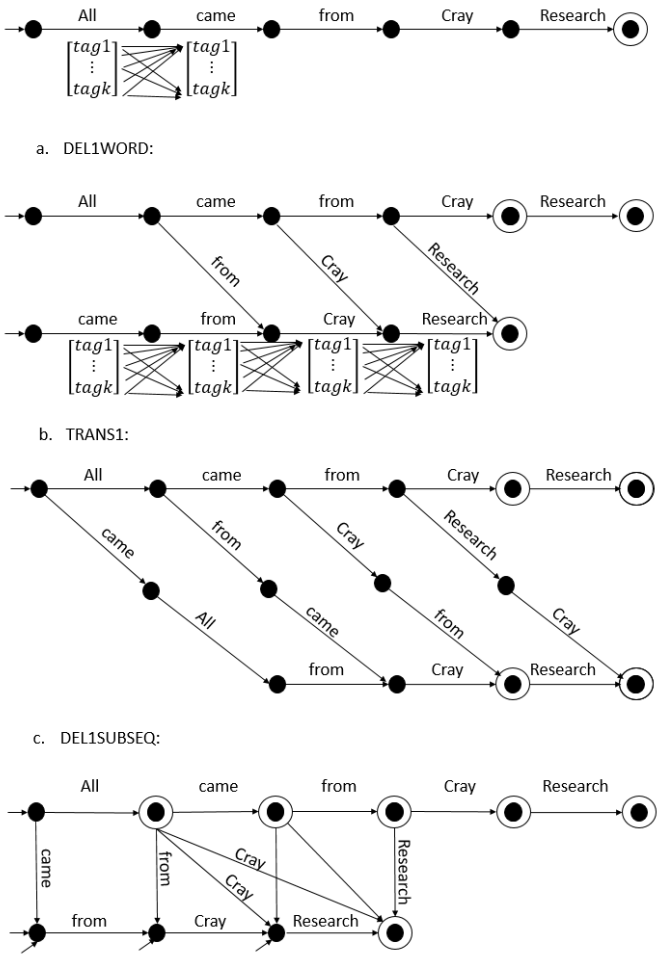


Fig. 2. Lattice structures of the sentence *"All came from Cray Research"* and its negative samples that are generated by different perturbation methods: deletion of a word (DEL1WORD), swapping of two words in a sequence (TRANS1), deletion of a word sequence in a sentence (DEL1SUBSEQ).

where $\alpha_t(j)$ is estimated recursively as follows:

$$\alpha_t(j) = \sum_{j=1}^{N} \alpha_{t-1}(i) a_{ij} b_j(w_t) \tag{18}$$

Here, the initial probability in the first time step is defined as given below:

$$\alpha_1(j) = a_{0j} b_j(w_1) \tag{19}$$

a_{ij} is the transition probability between the tags i and j, and $b_j(w_t)$ is the probability of the word w_t being emitted at time t from state j. Similarly, the backward probability is defined as follows:

$$\beta_t(i) = p(w_{t+1}, w_2, \cdots, w_T | y_t = i, \vec{\theta}) \tag{20}$$

This is the probability of seeing the words $w_1, w_2, \cdots, w_T$ given that we are in tag i at time t and it is estimated as follows:

$$\beta_t(i) = \sum_{j=1}^{N} \beta_{t+1}(j) a_{ij} b_j(w_{t+1}) \tag{21}$$

where the last time step for the last word in the sentence is defined as follows:

$$\beta_T(i) = a_{i(T+1)} \tag{22}$$

Both forward and backward probabilities are used to estimate the probability of a sentence at any time t as follows:

$$P(s_i|\vec{\theta}) = \sum_{j=1}^{N} \alpha_t(j)\beta_t(j) \tag{23}$$

3.2.2 Parameter Estimation. The log likelihood of data with an additional regularization term is defined as the objective function to be maximized:

$$L_N(\theta) = \sum_{s_i \in D} \log \frac{\sum_{y \in Y} u(s_i, y|\vec{\theta})}{\sum_{(s^*, y^*) \in N(s_i) \times Y} u(s^*, y^*|\vec{\theta})} - \frac{1}{\sqrt{2\pi\sigma^2}} \sum_k \frac{\theta^2}{\sigma^2} \tag{24}$$

where σ is the regularization parameter. For the estimation of the parameters for a maximum likelihood solution, a gradient descent algorithm called LBFGS-B [13] is used.

There are various types of algorithms proposed for the optimization problem in machine learning. Iterative scaling [32] and gradient-based optimization algorithms [36] are some of the well-known optimization algorithms. Minka [36] shows that conjugate gradient algorithms perform better than the iterative scaling methods for the logistic regression models. BFGS (Broyden-Fletcher-Goldfarb-Shanno) algorithm [22] is another gradient method frequently used for log-linear models, which is based on Quasi-Newton optimization method. Although it relies on the gradients at each learning step, it makes an intelligent choice for the search direction. L-BFGS algorithm [2, 56], is especially designed for limited memory when there are millions of parameters to be estimated and the algorithm is guaranteed to converge to a global optimal value. LBFGS [56] algorithm is considered to be the most efficient optimization algorithm for training log linear models [33] and conditional random fields (CRFs) [46]. LBFGS-B [13, 56] algorithm is an extension of LBFGS [33] that allows to define bounds on the variables in the model. Based on the other log-linear models, we decided to use the LBFGS-B algorithm for the optimization of our log-linear model since it is considered to be the most efficient optimization algorithm for log-linear models with a huge number of parameters.

The partial derivative with respect to the jth feature weight θ_j is computed as given below:

$$\begin{aligned} \frac{\partial L_N}{\partial \theta_j} &= \sum_{s_i \in D} E_\theta[f_j|s_i] - E_\theta[f_j|N(s_i)] \\ &= \sum_{s_i \in D} \left[\frac{\sum_{y \in Y} f_j(s_i, y,) \cdot u(s_i, y)}{\sum_{y \in Y} u(s_i, y)} - \frac{\sum_{(s^*, y^*) \in N(s) \times Y} f_j(s^*, y^*) \cdot u(w^*, y^*)}{\sum_{(s^*, y^*) \in N(s_i) \times Y} u(s^*, y^*|\vec{\theta})} \right] - \frac{\theta_j}{\sigma^2} \end{aligned}$$

where f_j is either a tag transition feature or an emission feature. Let $\varrho_t(i,j)$ be the probability of being in state i in time t and being in state j in time $t+1$ within the sentence s_k:

$$\varrho_t(i,j) = \mathbb{E}_\theta(y_t = i, y_{t+1} = j|s_k) \quad (25)$$

$$= \alpha_t(i) \times \hat{a}_{ij} \times \hat{b}_j(w_{t+1}) \times \beta_{t+1}(j) \quad \forall t,i,j \quad (26)$$

For the expected probability of the transition $\hat{a_{ij}}$, we sum the expected value of the transition i-j over all t for all sentences in D and divide it by the probability of the sentence to normalize both for the original sentence and the negative sentences:

$$\hat{a}_{ij} = \sum_{s_i \in D} \left[\frac{\sum_{t=1}^{T-1} \varrho_t(i,j)}{\sum_{k=1}^{N} \alpha_T(k)} - \sum_{(s^*,y^*) \in N(s_i) \times Y} \left(\frac{\sum_{t=1}^{T-1} \varrho_t(i,j)}{\sum_{k=1}^{N} \alpha_T(k)} \right) \right] \quad (27)$$

The expected emission probability with respect to one of the emission weights is estimated analogously. We define the expected probability of being in state j at time t as follows:

$$\upsilon_t(j) = \mathbb{E}_\theta(y_t = j|s) \quad (28)$$

$$= \alpha_t(j) \times \beta_t(j) \quad \forall t,j \quad (29)$$

The expected probability of observing w_k is estimated by summing over all t that emits the word w_k and it is normalized by the probability of the sentence under all possible tag paths:

$$b_j(\hat{w_k}) = \sum_{s_i \in D} \left[\frac{\sum_{t=1 s.t. O_t = w_k}^{T} \upsilon_t(j)}{\sum_{k=1}^{N} \alpha_T(k)} - \sum_{(s^*,y^*) \in N(s_i)} \left(\frac{\sum_{t=1 s.t. O_t = w_k}^{T} \upsilon_t(i,j)}{\sum_{k=1}^{N} \alpha_T(k)} \right) \right] \quad (30)$$

Here, a function $(O_t = w_k)$ checks whether the word equals to w_k.

3.3 The Cascaded PoS Tagging Framework

First, we train the Bayesian model to have the initial tags that are learned partially. However, those tags are not accurate enough yet and the probabilities of the tag-word emissions and the tag-tag transitions are estimated superficially. Then, we use the predicted tags in order to initialize the parameters of the log-linear model.

During initialization, in order to compute the weights that correspond to the θ values of the transition features (tag bigrams) and the emission features of the log linear model, we use the following equations respectively:

$$p(t_i|t_{i-1}) = \frac{n_{(t_{i-1},t_i)}}{n_{(t_{i-1})}} \quad (31)$$

$$p(w_i|t_i) = \frac{n_{(t_i,w_i)}}{n_{(t_i)}} \quad (32)$$

While calculating the probabilities of the transition and emission features, we assume that the predicted PoS tags are the gold tags, which is partially true in the first step of the cascaded algorithm. Here, we do not apply smoothing since we use the same dataset to further improve the tags that are learned partially in the Bayesian model. Therefore, there will not be any zero probabilities in the log linear model initially.

Once the transition and emission probabilities are learned by the Bayesian model, those probabilities estimated by Equation 31 and Equation 32 are used to initialize the parameters θ in the log linear model, defined in Equation 11.

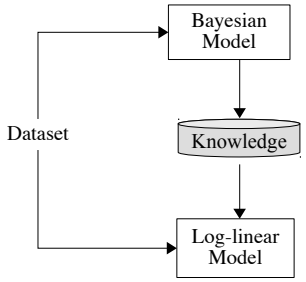


Fig. 3. The overall cascaded tagging framework.

Therefore, the model is trained further to improve tagging with the log linear model. The overview of the algorithm is given in Figure 3. One unsupervised model can learn to some degree, however the results are improved substantially when two different unsupervised models are combined in a single cascaded framework.

4 EXPERIMENTS & RESULTS

We present the experimental results by comparing our cascaded learning framework with the baseline log-linear model and the Bayesian model[1]. Additionally, we compare our results with two other models: Anchor HMM [51] and Brown Clustering [11]. As mentioned in Section 2, both Anchor HMM model and Brown Clustering are unsupervised models. Anchor HMM model is learned through a negative matrix factorization [3] and Brown clustering is based on language modeling, where each word is defined with its PoS tag cluster and those clusters are learned by maximum likelihood estimation.

4.1 Datasets

We performed all experiments on English and Turkish. For English, we used either the first 12K words or the full dataset of the WSJ Penn Treebank [34] that involves 17 tags [48] and for Turkish we used METU Treebank [39] that contains 5.620 sentences and 53.798 word tokens with 12 tags [7].[2]

4.2 Evaluation Metrics

Evaluating unsupervised PoS tagging is comparably difficult than that of supervised PoS tagging because the output of unsupervised models are not the actual PoS tags but state identifiers. Hence, using accuracy as evaluation method in unsupervised PoS tagging is impossible. Studies [23, 24, 30] on this task explored a variety of methodologies to address this issue. The most common metrics to evaluate unsupervised PoS tagging are clustering evaluation metrics. Those metrics are many-to-one, one-to-one, normalized mutual information (NMI), and variation of information (VI).

The main challenge in the evaluation of such unsupervised models is the matching of the predicted PoS tags to the gold PoS tags. Many-to-one [30] maps each result tag proposed by the unsupervised algorithm to the gold standard tag in the test set that has the maximum number of common words with the result tag. A greedy search is applied for mapping each result tag to a gold tag. One-to-one [27] is similar to many-to-one, however, it restricts the number of

[1]The implementation of both models will be publicly available if the article gets accepted.
[2]12 tags in Turkish: Noun,Adj,Adv,Conj,Det,Interj,Ques,Verb,Postp,Num,Pron,Punc

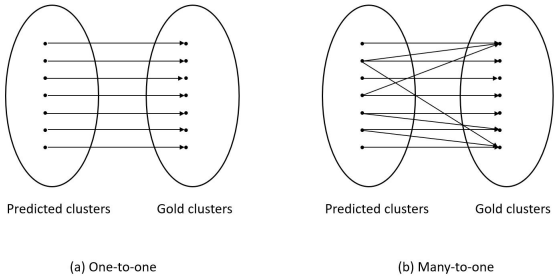


Fig. 4. Example clusters for (a) One-to-one and (b) Many-to-one

assigned tags to only 1 gold tag for each result tag. Example mappings between the predicted tags and the gold tags according to one-to-one and many-to-one are given in Figure 4.

NMI and VI are information theoretic measures that evaluate the divergence between two clusterings. NMI is an information-theoretic measure used for determining the quality of the clusters. It measures the overlapping of the result clusters on the gold clusters. NMI is 0 if there is no information shared between two clusterings, and NMI is 1 if two clusterings are the same. Therefore, NMI [52] gives the mutual dependence between two clusterings: C_r (result clustering) and C_g (gold clustering). The NMI score is computed as follows:

$$NMI(C_r, C_g) = \frac{I(C_r, C_g)}{\sqrt{H(C_r)H(C_g)}} \tag{33}$$

where H denotes the entropy and I denotes the mutual information between the two clusterings. Finally, VI [35] gives the distance between two clusterings:

$$VI(C_r, C_g) = H(C_r|C_g) + H(C_g|C_r) \tag{34}$$

where the conditional entropy is calculated mutually for both clusterings.

4.3 Results

We performed the experiments to analyze the effect of the variables in both Bayesian and log-linear model. Initially, we performed experiments to show the impact of the perturbation method employed in contrastive estimation.

First, we tested with different combinations of sentence perturbation methods. To this end, we trained both models on English dataset. In these experiments, we set $\sigma^2 = 10^{-2/3}$. The many-to-one accuracy scores are given in Table 1. The results show that when all perturbation methods are used together, more negative sentences are generated and more probability mass could be shifted towards the positive sentences, thereby we obtain the highest score, which is 80.88% many-to-one accuracy in the joint setting.

We applied a grid search method for hyperparameter tuning for the Bayesian model [24]. For that purpose, we assigned different values for α and β: $\alpha \in \{1, 0.1\}$ and $\beta \in \{0.003, 0.001, 0.03\}$. For the selection of these hyperparameters, we make use of the results presented by Goldwater and Griffiths [24]. Results obtained from the Bayesian model for different values of hyperparameters are represented in Table 2. Since we obtain the highest scores for $\alpha = 0.03$, $\beta = 0.1$, we use the output of that setting in the Bayesian HMM model as the input of the log-linear model.

Table 1. English PoS tagging results for different combination of perturbation methods

Model	Many-to-One
Cascaded Model DEL1WORD	54.97
Cascaded Model TRANS1	55.19
Cascaded Model DEL1SUBSEQ	52.93
Cascaded Model DEL1WORD & TRANS1	61.45
Cascaded Model DEL1WORD & DEL1SUBSEQ	64.54
Cascaded Model TRANS1 & DEL1SUBSEQ	69.61
Cascaded Model DEL1WORD & TRANS1 & DEL1SUBSEQ	**80.88**

Table 2. Many-to-one scores obtained from different hyperparameter values with Bayesian model for Turkish and English PoS tagging

	Value of α	Value of β 0.003	0.001	0.03
Turkish	1	54.92	55.81	55.51
	0.1	55.44	55.31	**56.79**
English	1	32.85	32.15	33.30
	0.1	32.96	32.11	**35.04**

Table 3. English many-to-one scores obtained by different values of σ parameter

σ^2	$10^{-2/3}$	$10^{-1/3}$	$10^{1/3}$	$10^{2/3}$
Cascaded Model	**81.80**	81.40	54.97	55.9

Table 4. PoS tagging results for English

Model	Many-to-One	One-to-One	NMI	VI
Bayesian HMM	35.04	21.91	0.11	5.88
Log-linear (Uniform)	35.42	23.89	0.17	**3.15**
Log-linear	33.37	24.48	0.15	**3.15**
Cascaded Model	**81.80**	30.68	0.06	4.21
Anchor HMM [51][1]	52.36	-	-	-
Brown Clustering [11] [2]	63.42	**42.70**	**0.44**	3.98

[1]Anchor HMM: https://github.com/karlstratos/anchor

[2]Brown Clustering: http://www.cs.berkeley.edu/~pliang/software/brown-cluster-1.2.zip (Percy Liang)

We also performed experiments to analyze the impact of different values of the regularization term in the log linear model. We used a 0-mean diagonal Gaussian prior with $\sigma^2 \in \{10^{-2/3}, 10^{-1/3}, 10^{1/3}, 10^{2/3}\}$. The English results are given in Table 3. The highest many-to-one score is obtained by $\sigma^2 = 10^{-2/3}$. The results for $\sigma^2 = 10^{-2/3}$ and $\sigma^2 = 10^{-1/3}$ are almost the same and worse compared to smaller values of σ.

In order to observe the impact of the initialization of the parameters in the log linear model, we experimented with different initializations of the model. For that purpose, we used uniform and random initializations in addition to the aforementioned pre-trained Bayesian weights. For the random weights, we assigned values between 0 and 1000 randomly for each feature in the log linear model. As for the uniform weights, we assigned the same fixed values such as 0.01 and 0.001 for the initial weights in the log linear model. When uniform and random weights were used as the initial weights of model, the model did not to converge to a stable state.

Table 5. PoS tagging results for Turkish

Model	Many-to-One	One-to-One	NMI	VI
Bayesian HMM	56.79	27.31	0.23	4.79
Log-linear (Uniform)	37.15	24.56	0.18	3.15
Log-linear	33.45	22.18	0.14	3.15
Cascaded Model	**75.45**	**37.21**	0.06	**2.72**
Anchor HMM [51] [1]	62.18	-	-	-
Brown Clustering [11] [2]	60.13	30.69	**0.29**	4.67

[1]Anchor HMM: https://github.com/karlstratos/anchor

[2]Brown Clustering: http://www.cs.berkeley.edu/~pliang/software/brown-cluster-1.2.zip (Percy Liang)

The overall English results for different evaluation metrics are given in Table 4. In the experiments for the cascaded learning model, DEL1WORD, TRANS1 and DEL1SUBSEQ are used together as the perturbation methods and we set $\sigma^2 = 10^{-2/3}$. When only the Bayesian model is used in an unsupervised setting unlike the way it was reported in the original paper (without using any tagging dictionary), we obtained a many-to-one accuracy of 35.04%. We obtained 33.37% and 35.42% many-to-one accuracy when we used the log-linear model with random and uniform initialization respectively. However, we obtained a many-to-one accuracy of 81.80% when we combined the two models by transferring the predicted tags from the Bayesian model to the log linear model. It is clearly seen that, using cascaded learning between two unsupervised models makes a significant improvement on the results for English. Compared to Brown Clustering [11] and Anchor HMM [51], we also obtained the highest many-to-one accuracy among these two models. The one-to-one accuracy score also improves upon the Bayesian and log-linear models. We obtained a one-to-one accuracy of 30.68% from the cascaded model, whereas Bayesian model gives an accuracy of 21.91% and the log-linear model gives an accuracy of 23.89% with the uniform initialization and 24.48% with the random initialization. The cascaded model performs better than both models, however the Brown clustering outperforms all models with a one-to-one accuracy of 42.70%. Our cascaded model does not perform as well as other models regarding the NMI measure. We obtain an NMI measure of 0.06, whereas Brown Clustering gives an NMI measure of 0.44. Both Bayesian and log-linear models also perform better than the cascaded model according to the NMI measure. NMI measures homogeneity and completeness similar to V-measure. Homogeneity is maximized when each cluster contains items of a single class and completeness aims to place the items of a single class into a single cluster. When we analyze the result clusters, it seems that some tags are scattered through various clusters. For example, nouns are usually confused with other tags and therefore they are distributed over different clusters, which lowers the completeness score. The cascaded model performs better than the Bayesian model regarding the VI score. We obtain a VI score of 4.21, whereas the Bayesian model gives a VI score of 5.88. Since VI measures the distance between two clusterings, a good clustering gives a lower VI score.

In the Turkish experiments for the cascaded model, DEL1WORD, TRANS1 and DEL1SUBSEQ are used as perturbation method and σ^2 is fixed to $10^{-2/3}$. The Turkish results are given in Table 5. When only the Bayesian model is used in an unsupervised setting unlike the way it was reported in the original paper (without using any tagging dictionary), we obtained a many-to-one accuracy of 56.79%. Result of the Bayesian model for Turkish is higher than English due to the size of data. We obtained 33.45% and 37.15% many-to-one accuracy when we used the log-linear model with random and uniform initialization respectively. We obtained a many-to-one accuracy of 75.45% when we combined the two models by transferring the learned knowledge from the Bayesian setting to the log linear setting. We improved the many-to-one

Table 6. English results obtained from the full and the partial datasets

Model	Many-to-One	One-to-One	NMI	VI
Bayesian HMM	42.79	34.75	**0.37**	4.45
Cascaded Model (all words in WSJ)	**74.38**	30.03	0.11	4.38
Cascaded Model (12k in WSJ)	74.32	**30.87**	0.12	**4.32**

accuracy with more than 20% compared to the original Bayesian model and we improved the many-to-one accuracy with more than 40% compared to the baseline log linear model. The cascaded model also outperforms all other models based on the one-to-one accuracy with a score of 37.21%, whereas the highest among the other models is obtained by Brown clustering with an accuracy of %30.69. When it comes to NMI measure, Brown clustering outperforms other models with NMI measure of 0.29 and the Bayesian model comes the second for the NMI measure. The cascaded model gives a NMI measure of 0.06 similar to English results. When we analyze the result clusters, again it seems that some tags are scattered through various clusters, which lowers the completeness score. For example, nouns and adjectives are usually confused with other. Regarding the VI measure, the cascaded model outperforms all models with a distance of 2.72, which shows that the distance between our result clusters and the gold clusters is much smaller compared to the results of the other models.

In order to analyze the impact of the corpus size on the cascaded model, we trained the Bayesian HMM Model [24] with all the words in the WSJ Penn Treebank. Once the transition and emission probabilities are learned by the Bayesian model, the parameters are used to initialize the log-linear model. Table 6 shows the results obtained from the Bayesian model on the full corpus and the results obtained from the cascaded model on the full and the 12K portion of the corpus, which are both initialized by the Bayesian model trained on the full corpus. The many-to-one accuracy and VI are slightly better in the full corpus training, however one-to-one accuracy and NMI are slightly lower in the full corpus training. Nevertheless, the scores are similar for both corpora size and this shows that the corpus size does not have a high impact on the model training, which allows for low-resource training as well.

4.4 Error Analysis

We performed a qualitative error analysis to understand the common errors in the results. One difficulty of the tagging task is the ambiguity of the PoS tags. A word may have different syntactic roles in different contexts. For example, the word *sets* is tagged as a noun or verb; the word *right* is tagged as noun or adjective in English depending on the context. Similarly, the word *nasıl* (means *how*) is tagged as an adverb, adjective, or verb in Turkish depending on its context.

Some examples obtained from the English results are given in Table 7. The results obtained from the Bayesian model, log-linear model with random initialization, and the cascaded model are given along with their gold tags. The table gives various contexts of the word *right*, where in two different contexts it is used as an adjective, and in the latter it is used as a noun. Only contexts of five-grams are given to reduce the space in the tables. The results show that although Bayesian and log-linear model tag the word *right* as either verb or noun in the first context, where as the cascaded model correctly tags it as an adjective. In the second context, *to right ones*, although log-linear model also tags the word *right* as adjective correctly, the tags of the context words are not predicted correctly by the log linear model. Only cascaded model tags the sequence correctly apart from the tag of the preposition *on*. In another context where the word *right* is used as a noun, all models tag it correctly. However, the contextual tag sequence obtained from the cascaded model is closer to the gold tag sequence compared to other models. Since a larger context is used in the

Table 7. Some English results obtained from the Bayesian, log-linear with random initialization, and the cascaded model are given along with their gold tags. Here, the example contexts with a window size of five are given for the word *right*.

Model	Example (w_{t-2}, w_{t-1}, w_t, w_{t+1}, w_{t+2})
Bayesian	's/POS the/DET right/V time/N ./ENDPUNC
Log-linear	's/N the/N right/N time/N ./N
Cascaded	's/PREP the/N right/ADJ time/N ./ENDPUNC
Gold	's/V the/DET right/ADJ time/N ./ENDPUNC
Bayesian	answers/N to/TO right/V ones/N on/TO
Log-linear	answers/N to/POS right/ADJ ones/N on/ADJ
Cascaded	answers/N to/TO right/ADJ ones/N on/TO
Gold	answers/N to/TO right/ADJ ones/N on/PREP
Bayesian	woman/N 's/TO right/N to/TO choose/N
Log-linear	woman/N 's/V right/N to/ADJ choose/0
Cascaded	woman/N 's/TO right/N to/TO choose/V
Gold	woman/N 's/POS right/N to/TO choose/V

log-linear model, with a better initialization, the cascaded model can learn the tag sequences better than both Bayesian and log-linear model.

Some examples obtained from the Turkish results are given in Table 8. The results obtained from the Bayesian model, log-linear model with random initialization, and the cascaded model are again given along with their gold tags. The table gives various contexts of the word *nasıl* (meaning *how*), where in two contexts it is used as an adverb and in two other contexts it is used as an adjective. In the first context, Bayesian and log-linear model tag the word *nasıl* incorrectly, whereas the cascaded model tags it correctly as adverb in the context *hatayı nasıl yapar* (meaning *how can he/she make this mistake*). In the second context, again Bayesian and log-linear model tag the word *nasıl* as noun incorrectly, whereas the cascaded model tags it correctly as adverb in the context *ona nasıl söyleriz* (meaning *how can we tell him/her*). In the third context, the Bayesian model and the log-linear model again incorrectly tag the word *nasıl* as a verb and noun respectively, whereas the cascaded model tags it correctly as adjective in the context *durumu nasıl sahi* (meaning *how is her/his status indeed*). Finally, in the last context, the Bayesian model correctly tags the word *nasıl* as adjective, but the log-linear model incorrectly tags it as noun. The cascaded model again correctly tags it as adjective in the context *nasıl bir viski* (meaning *what kind of a whisky*). When the tag sequences in all examples are examined, we can see that the cascaded model can learn the tag sequences better compared to other models. Therefore, the Turkish results are coherent with the English results.

More English results are given in Table 9. Due to the high frequency of the tags *LPUNC* and *RPUNC* (punctuation), sometimes our model tends to assign the punctuation tag to other categories incorrectly. For example, *said* is tagged as a punctuation incorrectly and *led* is tagged also as punctuation incorrectly. However, still the cascaded model can predict the punctuation more correctly compared to other model. For example, the comma is tagged correctly as punctuation by the cascaded model, whereas it is either tagged as a verb or adverb by the Bayesian and log-linear model respectively. Nouns are very frequent in the dataset and it might be still confused with the verbs in the cascaded model. However, when we examine the clusters in the Bayesian model once the mapping between result and gold tags is applied, it is seen that nouns are spread over many clusters in the Bayesian model. However, nouns are spread over fewer clusters in the cascaded model. It is seen that this type of clustering is also observed in the in other supervised models [24, 27]. We also observed from the results that adverbs and adjectives are usually confused with each other in the cascaded model, which is also a common error among other models [24]. In short, most of the tags in the Bayesian model are spread

Table 8. Some Turkish results obtained from the Bayesian, log-linear with random initialization, and the cascaded model are given along with their gold tags. Here, the example contexts with a window size of five are given for the word *nasıl*.

Model	Example (w_{t-2}, w_{t-1}, w_t, w_{t+1}, w_{t+2})
Bayesian	bu/N hatayı/N nasıl/ADJ yapar/N "/PUNC
Log-linear	bu/N hatayı/N **nasıl/N** yapar/PUNC "/N
Cascaded	bu/N hatayı/N nasıl/ADV yapar/N "/N
Gold	bu/DET hatayı/N nasıl/ADV yapar/V "/PUNC
Bayesian	gerçeği/ADJ ona/N nasıl/N söyleriz/V bilmiyorum/N
Log-linear	gerçeği/PUNC ona/Pron **nasıl/N** söyleriz/N bilmiyorum/PUNC
Cascaded	gerçeği/N ona/Pron nasıl/ADV söyleriz/N bilmiyorum/N
Gold	gerçeği/N ona/Pron nasıl/ADV söyleriz/V bilmiyorum/V
Bayesian	teyzenizin/N durumu/N nasıl/V sahi/N ?/PUNC
Log-linear	teyzenizin/N durumu/N nasıl/N sahi/N ?/V
Cascaded	teyzenizin/DET durumu/N nasıl/ADJ sahi/ADV ?/PUNC
Gold	teyzenizin/N durumu/N nasıl/AD sahi/ADV ?/PUNC
Bayesian	START/<s> START/<s> nasıl/ADJ bir/N viski/ADJ
Log-linear	START/<s> START/<s> nasıl/N bir/N viski/N
Cascaded	START/<s> START/<s> nasıl/ADJ bir/N viski/N
Gold	START/<s> START/<s> nasıl/ADJ bir/DET viski/N

over many clusters, however in the cascaded model they are aggregated in fewer clusters. The improvement in the cascaded model is due to a wider context that is considered during tagging. In Bayesian model, only trigram contexts are considered, whereas the contexts could be a sentence wide in the log-linear model due to the lattice structures although only bigram features are employed in the model. Therefore, the improvement in the cascaded model is much more visible in the longer sentences.

The confusion matrix obtained from the English results are given in Figure 5 for three models. The columns refer to the gold PoS labels and the columns correspond to the predicted PoS tags, which are learned from the unsupervised models (Bayesian, Log-linear and Cascaded). As seen from the figure, most of the tags are almost uniformly distributed over different clusters in both Bayesian and log-linear model. The clusters are better defined in the cascaded model and most of the frequent tags such as DET, PREP, ENDPUNC, VBN, ADV, ADJ are usually correctly predicted. The most erroneous tag is N for the nouns, which are scattered over many clusters. The confusion matrices also indicate that the Bayesian and the log-linear models are complementary. For example, adjectives scatter over a number of clusters in Bayesian model and log-linear model cannot assign the adjectives correctly in most cases. However, the cascaded model can correctly tag the adjectives in general. The same also applies to other clusters such as PREP, ENDPUNC, and DET.

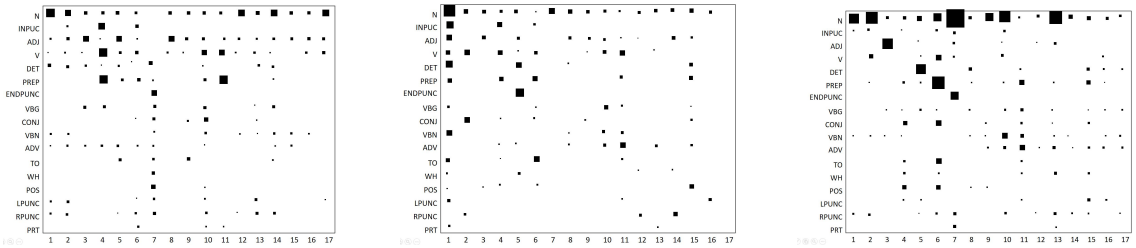
The confusion matrix obtained from the Turkish results are given in Figure 6 for three models. Similar to the confusion matrices built for the English results, the columns refer to the gold PoS labels and the columns correspond to the predicted PoS tags, which are learned from the unsupervised models (Bayesian, Log-linear and Cascaded) Most of the clusters are almost uniformly distributed in Bayesian and log-linear model analogously to English results. However, completeness improves in the cascaded model since each cluster has fewer number of items from different classes. For example, V, CONJ, PUNC are usually predicted correctly. However, nouns are scattered over many clusters similar to English results.

Table 9. More examples to English results obtained from the Bayesian, log-linear and the cascaded model along with their gold tag sequences.

Model	Example (w_{t-2}, w_{t-1}, w_t, w_{t+1}, w_{t+2})
Bayesian	”/V said/N Dr./DET Talcott/N ./ENDPUNC
Log-linear	”/ADJ said/ADJ Dr./N Talcott/N ./N
Cascaded	”/RPUNC said/INPUNC Dr./N Talcott/N ./ENDPUNC
Gold	"/RPUNC said/V Dr./N Talcott/N ./ENDPUNC
Bayesian	body/N ,/V Dr./V Mossman/V explained/V
Log-linear	body/N ,/ADV Dr./ADJ Mossman/ADJ explained/ADJ
Cascaded	body/N ,/INPUNC Dr./N Mossman/N explained/N
Gold	body/N ,/INPUNC Dr./N Mossman/N explained/V
Bayesian	START/<s> START/<s> Dr./N Talcott/N led/V
Log-linear	START/<s> START/<s> Dr./N Talcott/N led/N
Cascaded	START/<s> START/<s> Dr./ADJ Talcott/N led/INPUNC
Gold	START/<s> START/<s> Dr./N Talcott/N led/V
Bayesian	a/DET fraction/N of/PREP a/DET percentage/N point/N to/PREP 8.45/N
Log-linear	a/N fraction/N of/PREP a/N percentage/N point/N to/PREP 8.45/N
Cascaded	a/DET fraction/N of/PREP a/DET percentage/N point/N to/TO 8.45/ADJ
Gold	a/DET fraction/N of/PREP a/DET percentage/N point/N to/TO 8.45/ADJ
Bayesian	A./N for/PREP $/ADJ 295/ADJ million/ADJ ./ENDPUNC
Log-linear	A./N for/PREP $/ADJ 295/ADJ million/N ./ENDPUNC
Cascaded	A./N for/PREP $/ADJ 295/ADJ million/ADJ ./ENDPUNC
Gold	A./N for/PREP $/ADJ 295/ADJ million/ADJ ./ENDPUNC
Bayesian	Government/N officials/N said/V exports/N at/PREP the/DET end/N of/PREP
Log-linear	Government/N officials/N said/V exports/ADJ at/ADJ the/DET end/N of/N
Cascaded	Government/N officials/N said/V exports/N at/PREP the/DET end/N of/PREP
Gold	Government/N officials/N said/V exports/N at/PREP the/DET end/N of/PREP
Bayesian	That/ADJ got/N hard/V to/V take/V ,/INPUNC ”/RPUNC he/N
Log-linear	That/ADJ got/INPUNC hard/ADV to/TO take/N ,/ENDPUNC "/ENDPUNC he/N
Cascaded	That/N got/V hard/DV to/TO take/V ,/INPUNC ”/RPUNC he/N
Gold	That/W got/V hard/ADV to/TO take/V ,/INPUNC "/RPUNC he/N
Bayesian	is/V that/DET seymour/N is/N the/DET chief/N designer/N of/PREP the/DET
Log-linear	is/V that/N seymour/N is/ADJ the/N chief/N designer/N of/N the/N
Cascaded	is/V that/PREP Seymour/N is/V the/DET chief/ADJ designer/N of/TO the/DET
Gold	is/V that/PREP Seymour/N is/V the/DET chief/ADJ designer/N of/PREP the/DET

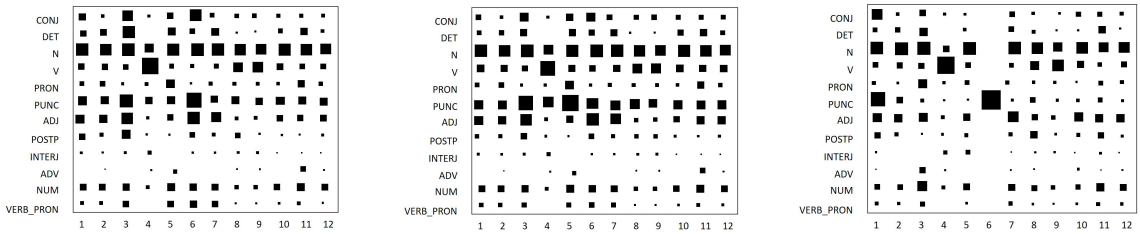
4.5 Discussion

Our results show that two different mathematical models can complement each other in a fully unsupervised setting. Every model has its own limitations. For example, the Bayesian model can tolerate sparse distributions with the prior distributions. However, in the absence of an available tagging dictionary, the model can hardly predict the underlying probability distributions for both transitions and emissions due to a very large search space that allows any word to be tagged with any label. With the help of the prior distributions, both transition and emission distributions are forced to be in Dirichlet form by preventing a skewed distribution. It can achieve up to 35% many-to-one accuracy for English and 56% many-to-one accuracy for Turkish. These scores show that Bayesian model can learn up to a level without using any external resource.



(a) Bayesian Model (b) Log-linear Model (c) Cascaded Model

Fig. 5. Confusion matrices obtained from the English results in (a) Bayesian Model and (b) Log-linear Model (c) Cascaded Model



(a) Bayesian Model (b) Log-linear Model (c) Cascaded Model

Fig. 6. Confusion matrices obtained from the Turkish results in the (a) Bayesian Model and (b) Log-linear Model (c) Cascaded Model

Log-linear models are very sensitive to the initialization of the parameters. This has been tested and the scores with a different initialization are given for both languages in the respective tables. The original log-linear model proposed by Smith and Eisner uses a tagging dictionary, and therefore the search space is reduced excessively during the training of the model. However, if the parameters are randomly initialized, the scores cannot go beyond 37% accuracy in the log linear model. Therefore, instead of using an external resource, we decided to update the parameters of the log-linear model with the deficient tags obtained from the trained Bayesian model.

The log-linear model considers all possible tag sequences in a sentence rather than modelling only bigram or trigram transitions. Therefore, the search space is much larger compared to the Bayesian model. Using a gradient-based optimization method that can be trained on an infinite number of parameters (i.e. LBFGS-B), can also tolerate this large search space.

Finally, we obtained a many-to-one accuracy of 81.80% for English and 75.45% for Turkish, which are both significantly higher than the single Bayesian model. This tells us that log-linear model can tolerate the wrong information while searching for the global optimum values of parameters by considering a larger context within sentences until it reaches to a stable situation.

To analyze the effects of embedding a fully supervised model into the cascaded model to initialize the parameters of an unsupervised model, we trained TnT tagger [9] (a fully supervised model) on the full Penn TreeBank dataset excluding the first 498 sentences which is used as a test set. We obtained %100.0 of many-to-one accuracy, 86.12% of one-to-one accuracy, and 1.012 of VI from the fully supervised model. Then, we initialized the log-linear model with the parameters learned by the supervised model. In other words, we replaced the Bayesian model with TnT tagged in

the cascaded framework. Finally, we obtained 100.0% of many-to-one accuracy, 86.14% of one-to-one accuracy, and 1.012 of VI. Although some scores have slightly improved, there is not a significant improvement on the scores of the supervised model as expected. Even finding the global maximum is not guaranteed in a log linear model because a log linear model with hidden variables is not globally concave. As it is stated by Smith and Eisner [48], the bias in the initialization of the parameters will affect the quality of the estimate and the performance of the method. Here, the results are not improved further if we use a supervised model for the initialization of the parameters, which is due to the final global maximum obtained in the fully supervised model, which cannot be improved further. All the randomness in the tags converge to a single point in the supervised model and this convergence does not affect the unsupervised model positively nor negatively.

5 CONCLUSION & FUTURE WORK

We present a cascaded learning model that combines the two unsupervised models to further improve both for the PoS tagging task. We used the fully Bayesian model proposed by Goldwater and Griffiths [24] to pre-train the model to obtain the predicted tags. We adopted the log-linear model proposed by Smith and Eisner [48] to further improve its results by initializing the model with the predicted parameters by the Bayesian model. The results show that when using two different unsupervised models, the results are improved significantly by up to 30% of accuracy compared to using only one of the unsupervised models.

Our contribution is twofold. On the other hand, the original models we adopted in this framework are not fully unsupervised and they both use tagging dictionaries to learn the PoS tags. However, in our framework we did not use any tagging dictionary and in a fully unsupervised setting, our results are even very competitive with the semi-supervised models.

Therefore, we released the need of an external resource which requires a great effort to be labelled. Normally, the so-called unsupervised models that use a tagging dictionary does not use a small tagging dictionary. Instead, they use a large amount of tagging dictionary. Such tagging dictionary exists in some languages. However, having such a tagging dictionary in low-resource languages is troublesome.

We believe that the accuracy can be slightly compromised for the sake of avoiding the need of a tagged corpus. This also automates the tagging process for any language, otherwise language-specific corpora should be also prepared for PoS tagging of another language.

Additionally, we leave extrinsic evaluation as a future goal by using the results of the cascaded model within a supervised NLP task such as shallow parsing or named entity recognition.

ACKNOWLEDGMENTS

We thank Salih Tuç for his contribution in the implementation of this project. This research was supported by the Scientific and Technological Research Council of Turkey (TUBITAK) with the project number EEEAG-115E464.

REFERENCES

[1] Daniel Andor, Chris Alberti, David Weiss, Aliaksei Severyn, Alessandro Presta, Kuzman Ganchev, Slav Petrov, and Michael Collins. 2016. Globally Normalized Transition-based Neural Networks. *arXiv preprint arXiv:1603.06042* (2016).

[2] Galen Andrew and Jianfeng Gao. 2007. Scalable Training of L1-regularized Log-linear Models. In *Proceedings of the 24th International Conference on Machine Learning (ICML '07)*. ACM, New York, NY, USA, 33–40.

[3] Sanjeev Arora, Rong Ge, Yonatan Halpern, David Mimno, Ankur Moitra, David Sontag, Yichen Wu, and Michael Zhu. 2013. A Practical Algorithm for Topic Modeling with Provable Guarantees. In *Proceedings of the 30th International Conference on Machine Learning (Proceedings of Machine Learning*

Research), Sanjoy Dasgupta and David McAllester (Eds.), Vol. 28. PMLR, Atlanta, Georgia, USA, 280–288. http://proceedings.mlr.press/v28/arora13.html

[4] Michele Banko and Robert C Moore. 2004. Part of speech tagging in context. In *Proceedings of the 20th international conference on Computational Linguistics*. Association for Computational Linguistics, 556.

[5] Taylor Berg-Kirkpatrick, Alexandre Bouchard-Côté, John DeNero, and Dan Klein. 2010. Painless unsupervised learning with features. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*. Association for Computational Linguistics, 582–590.

[6] Chris Biemann. 2006. Unsupervised part-of-speech tagging employing efficient graph clustering. In *Proceedings of the 21st International Conference on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics: Student Research Workshop*. Association for Computational Linguistics, 7–12.

[7] Necva Bölücü and Burcu Can. 2017. Joint PoS Tagging and Stemming for Agglutinative Languages. In *International Conference on Computational Linguistics and Intelligent Text Processing*. Springer, 110–122.

[8] Lubomir Bourdev and Jonathan Brandt. 2005. Robust Object Detection via Soft Cascade. In *Proceedings of the 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05) - Volume 2 - Volume 02 (CVPR '05)*. IEEE Computer Society, Washington, DC, USA, 236–243.

[9] Thorsten Brants. 2000. TnT – A Statistical Part-of-Speech Tagger. In *Sixth Applied Natural Language Processing Conference*. Association for Computational Linguistics, Seattle, Washington, USA, 224–231.

[10] Eric Brill. 1992. A simple rule-based part of speech tagger. In *Proceedings of the Third Conference on Applied Natural Language Processing*. Association for Computational Linguistics, 152–155.

[11] Peter F Brown, Peter V Desouza, Robert L Mercer, Vincent J Della Pietra, and Jenifer C Lai. 1992. Class-based n-gram models of natural language. *Computational linguistics* 18, 4 (1992), 467–479.

[12] S. Charles Brubaker, Jianxin Wu, Jie Sun, Matthew D. Mullin, and James M. Rehg. 2008. On the Design of Cascades of Boosted Ensembles for Face Detection. *International Journal of Computer Vision* 77, 1 (01 May 2008), 65–86.

[13] Richard H Byrd, Peihuang Lu, Jorge Nocedal, and Ciyou Zhu. 1995. A limited memory algorithm for bound constrained optimization. *SIAM Journal on Scientific Computing* 16, 5 (1995), 1190–1208.

[14] Eugene Charniak, Curtis Hendrickson, Neil Jacobson, and Mike Perkowitz. 1993. Equations for part-of-speech tagging. In *AAAI*, Vol. 11. 784–789.

[15] Danqi Chen and Christopher Manning. 2014. A fast and accurate dependency parser using neural networks. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. 740–750.

[16] Alexander Clark. 2003. Combining distributional and morphological information for part of speech induction. In *Proceedings of the Tenth Conference on European Chapter of the Association for Computational Linguistics-Volume 1*. Association for Computational Linguistics, 59–66.

[17] Ronan Collobert, Jason Weston, Léon Bottou, Michael Karlen, Koray Kavukcuoglu, and Pavel Kuksa. 2011. Natural language processing (almost) from scratch. *Journal of Machine Learning Research* 12, Aug (2011), 2493–2537.

[18] Doug Cutting, Julian Kupiec, Jan Pedersen, and Penelope Sibun. 1992b. A practical part-of-speech tagger. In *Proceedings of the third conference on Applied natural language processing*. Association for Computational Linguistics, 133–140.

[19] Douglass R Cutting, David R Karger, Jan O Pedersen, and John W Tukey. 1992a. Scatter/gather: A cluster-based approach to browsing large document collections. In *Proceedings of the 15th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*. ACM, 318–329.

[20] L De Lathauwer, B De Moor, J Vandewalle, and Blind Source Separation by Higher-Order. 1994. Singular value decomposition. In *Proceedings of EUSIPCO-94, Edinburgh, Scotland, UK*, Vol. 1. 175–178.

[21] David Elworthy. 1994. Does Baum-Welch Re-estimation Help Taggers?. In *Proceedings of the Fourth Conference on Applied Natural Language Processing (ANLC '94)*. Association for Computational Linguistics, Stroudsburg, PA, USA, 53–58.

[22] Roger Fletcher. 1987. *Practical Methods of Optimization; (2nd Ed.)*. Wiley-Interscience, New York, NY, USA.

[23] Jianfeng Gao and Mark Johnson. 2008. A comparison of Bayesian estimators for unsupervised Hidden Markov Model POS taggers. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, 344–352.

[24] Sharon Goldwater and Tom Griffiths. 2007. A fully Bayesian approach to unsupervised part-of-speech tagging. In *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics*. 744–751.

[25] Yu Gong, Xusheng Luo, Yu Zhu, Wenwu Ou, Zhao Li, Muhua Zhu, Kenny Zhu, Lu Duan, and Xi Chen. 2019. Deep Cascade Multi-Task Learning for Slot Filling in Online Shopping Assistant. *Proceedings of the AAAI Conference on Artificial Intelligence* 33 (07 2019), 6465–6472.

[26] Matthew R Gormley and Jason Eisner. 2013. Nonconvex global optimization for latent-variable models. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. 444–454.

[27] Aria Haghighi and Dan Klein. 2006. Prototype-driven learning for sequence models. In *Proceedings of the main conference on Human Language Technology Conference of the North American Chapter of the Association of Computational Linguistics*. Association for Computational Linguistics, 320–327.

[28] Geremy Heitz, Stephen Gould, Ashutosh Saxena, and Daphne Koller. 2009. Cascaded Classification Models: Combining Models for Holistic Scene Understanding. In *Advances in Neural Information Processing Systems 21*, D. Koller, D. Schuurmans, Y. Bengio, and L. Bottou (Eds.). Curran Associates, Inc., 641–648.

[29] Yong Jiang, Wenjuan Han, and Kewei Tu. 2016. Unsupervised neural dependency parsing. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*. 763–771.

[30] Mark Johnson. 2007. Why doesn't EM find good HMM POS-taggers?. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*.

[31] Matthieu Labeau, Kevin Löser, and Alexandre Allauzen. 2015. Non-lexical neural architecture for fine-grained POS tagging. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*. 232–237.

[32] John D. Lafferty, Andrew McCallum, and Fernando C. N. Pereira. 2001. Conditional Random Fields: Probabilistic Models for Segmenting and Labeling Sequence Data. In *Proceedings of the Eighteenth International Conference on Machine Learning (ICML '01)*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 282–289.

[33] Dong C Liu and Jorge Nocedal. 1989. On the limited memory BFGS method for large scale optimization. *Mathematical programming* 45, 1-3 (1989), 503–528.

[34] Mitchell P Marcus, Mary Ann Marcinkiewicz, and Beatrice Santorini. 1993. Building a large annotated corpus of English: The Penn Treebank. *Computational linguistics* 19, 2 (1993), 313–330.

[35] Marina Meilă. 2007. Comparing clusterings - an information based distance. *Journal of multivariate analysis* 98, 5 (2007), 873–895.

[36] Thomas Minka. 2001. Algorithms for maximum-likelihood logistic regression. (2001).

[37] Robert Moore. 2015. An improved tag dictionary for faster part-of-speech tagging. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*. 1303–1308.

[38] Karthik Narasimhan, Regina Barzilay, and Tommi Jaakkola. 2015. An unsupervised method for uncovering morphological chains. *arXiv preprint arXiv:1503.02335* (2015).

[39] Kemal Oflazer, Bilge Say, Dilek Zeynep Hakkani-Tür, and Gökhan Tür. 2003. Building a Turkish treebank. In *Treebanks*. Springer, 261–277.

[40] J.A. Perez-Ortiz and M.L. Forcada. 2001. Part-of-speech tagging with recurrent neural networks. (02 2001).

[41] Barbara Plank, Anders Søgaard, and Yoav Goldberg. 2016. Multilingual part-of-speech tagging with bidirectional long short-term memory models and auxiliary loss. *arXiv preprint arXiv:1604.05529* (2016).

[42] Adwait Ratnaparkhi. 1996. A maximum entropy model for part-of-speech tagging. In *Conference on Empirical Methods in Natural Language Processing*.

[43] Sujith Ravi and Kevin Knight. 2009. Minimized models for unsupervised part-of-speech tagging. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP: Volume 1-Volume 1*. Association for Computational Linguistics, 504–512.

[44] Henry Schneiderman. 2004. Feature-centric Evaluation for Efficient Cascaded Object Detection. In *Proceedings of the 2004 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'04)*. IEEE Computer Society, Washington, DC, USA, 29–36.

[45] Hinrich Schütze. 1993. Part-of-speech induction from scratch. In *Proceedings of the 31st Annual Meeting on Association for Computational Linguistics*. Association for Computational Linguistics, 251–258.

[46] Fei Sha and Fernando Pereira. 2003. Shallow Parsing with Conditional Random Fields. In *Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology - Volume 1 (NAACL '03)*. Association for Computational Linguistics, Stroudsburg, PA, USA, 134–141.

[47] Noah Smith. 2006. *Novel estimation methods for unsupervised discovery of latent structure in natural language text*. Ph.D. Dissertation.

[48] Noah A Smith and Jason Eisner. 2005. Contrastive estimation: Training log-linear models on unlabeled data. In *Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics*. Association for Computational Linguistics, 354–362.

[49] Noah A Smith and Jason Eisner. 2006. Annealing structural bias in multilingual weighted grammar induction. In *Proceedings of the 21st International Conference on Computational Linguistics and the 44th annual meeting of the Association for Computational Linguistics*. Association for Computational Linguistics, 569–576.

[50] Valentin I Spitkovsky, Hiyan Alshawi, and Daniel Jurafsky. 2010. From baby steps to leapfrog: How less is more in unsupervised dependency parsing. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*. Association for Computational Linguistics, 751–759.

[51] Karl Stratos, Michael Collins, and Daniel Hsu. 2016. Unsupervised part-of-speech tagging with anchor hidden markov models. *Transactions of the Association for Computational Linguistics* 4 (2016), 245–257.

[52] Alexander Strehl and Joydeep Ghosh. 2002. Cluster ensembles—a knowledge reuse framework for combining multiple partitions. *Journal of machine learning research* 3, Dec (2002), 583–617.

[53] Nicola Ueffing and Hermann Ney. 2003. Using POS information for statistical machine translation into morphologically rich languages. In *Proceedings of the tenth conference on European chapter of the Association for Computational Linguistics-Volume 1*. Association for Computational Linguistics, 347–354.

[54] Peilu Wang, Yao Qian, Frank K Soong, Lei He, and Hai Zhao. 2015. Part-of-speech tagging with bidirectional long short-term memory recurrent neural network. *arXiv preprint arXiv:1510.06168* (2015).

[55] Ming Yan, Jiangnan Xia, Chen Wu, Bin Bi, Zhongzhou Zhao, Ji Zhang, Luo Si, Rui Wang, Wei Wang, and Haiqing Chen. 2018. A Deep Cascade Model for Multi-Document Reading Comprehension.

[56] Ciyou Zhu, Richard H Byrd, Peihuang Lu, and Jorge Nocedal. 1997. Algorithm 778: L-BFGS-B: Fortran subroutines for large-scale bound-constrained optimization. *ACM Transactions on Mathematical Software (TOMS)* 23, 4 (1997), 550–560.